AMENDMENTS TO THE CLAIMS

Kindly amend claims 1, 6, 8-10, 12, 15, 22, 24-25, 28, 31-32, 36, 39, 45, and 48, and add new claims 51-70, as shown in the following listing of claims. The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (currently amended) A branch prediction apparatus in a processor including address selection logic for providing a fetch address to an instruction cache, the fetch address used to select lines of the instruction cache, the apparatus comprising:

   first and second branch predictors, for providing first and second target address predictions of ~~an unconditional~~ branch instruction to the address selection logic;

   instruction decode logic, configured to receive and decode said ~~unconditional~~ branch instruction and to generate a type thereof; and

   branch control logic, configured to control the address selection logic to select said first prediction as the fetch address, said first prediction selecting a first line of the instruction cache;

   wherein said branch control logic is further configured to subsequently selectively control the address selection logic, based on said branch instruction type, to select said second prediction as the fetch address, said second prediction selecting a second line of the instruction cache.

2. (original) The apparatus of claim 1, further comprising:

   comparison logic, coupled to said first and second branch predictors, for comparing said first and second target address predictions.

3. (original) The apparatus of claim 2 wherein said type includes a specification of whether said branch instruction is a return type branch instruction.

4. (original) The apparatus of claim 3, wherein said branch control logic controls the address selection logic to select said second target address prediction if said branch instruction type is a return instruction and said first and second predictions miscompare.

5. (original) The apparatus of claim 4, wherein said second branch predictor comprises a call/return stack for providing said second target address prediction of said return instruction.

6. (currently amended) The apparatus of claim 2, wherein said type includes a specification of whether said ~~unconditional~~ branch instruction is a program counter-relative type branch instruction.

7. (original) The apparatus of claim 6, wherein said branch control logic controls the address selection logic to select said second target address prediction if said branch instruction type is a program counter-relative branch instruction and said first and second predictions miscompare.

8. (currently amended) The apparatus of claim 7, wherein said second branch predictor comprises an arithmetic unit for calculating said second target address prediction based on an instruction pointer of said ~~unconditional~~ branch instruction.

9. (currently amended) The apparatus of claim 8, wherein said arithmetic unit calculates said second target address prediction using said instruction pointer of said ~~unconditional~~ branch instruction.

10. (currently amended) The apparatus of claim 2, wherein said type includes a specification of whether said ~~unconditional~~ branch instruction is a direct type branch instruction.

11. (original) The apparatus of claim 10, wherein said branch control logic controls the address selection logic to select said second target address prediction if said branch instruction type is a direct branch instruction and said first and second predictions miscompare.

12. (currently amended) The apparatus of claim 2, wherein said type includes a specification of whether said ~~unconditional~~ branch instruction is an indirect type branch instruction.

13. (original) The apparatus of claim 12, wherein said branch control logic controls the address selection logic not to select said second target address prediction if said branch instruction type is an indirect branch instruction.

14. (original) The apparatus of claim 13, wherein said second branch predictor comprises a branch target buffer for caching branch target addresses of previously executed indirect branch instructions.

15. (currently amended) The apparatus of claim 2, ~~wherein said first and second predictors are also configured to provide said first and second target address predictions of a conditional branch instruction to the address selection logic,~~ wherein said type includes a specification of whether said branch instruction is a conditional type branch instruction.

16. (original) The apparatus of claim 15, wherein said branch control logic controls the address selection logic to select said second target address prediction if said branch instruction type is a conditional branch instruction and said first and second predictions miscompare.

17. (original) The apparatus of claim 15, wherein said first and second predictors provide first and second direction predictions of said conditional branch instruction to said branch control logic for predicting whether said conditional branch instruction will be taken.

18. (original) The apparatus of claim 17, further comprising:

second comparison logic, coupled to said first and second branch predictors, for comparing said first and second direction predictions of said conditional branch instruction.

19. (original) The apparatus of claim 18, wherein said branch control logic controls the address selection logic to select an instruction pointer of a next sequential instruction to said conditional branch instruction as the fetch address if said second direction prediction predicts said conditional branch instruction will not be taken.

20. (original) The apparatus of claim 19, wherein said branch control logic controls the address selection logic to select said next sequential instruction pointer if said second direction prediction predicts said conditional branch instruction will not be taken and said first and second direction predictions miscompare.

21. (original) The apparatus of claim 2, wherein said branch control logic subsequently selectively controls the address selection logic based on said branch instruction type to select said second prediction as the fetch address if said first and second predictions do not match.

22. (currently amended) The apparatus of claim 1, wherein said ~~unconditional~~ branch instruction type comprises an Intel IA-32 instruction set branch instruction type.

23. (original) The apparatus of claim 1, wherein said first branch predictor receives the instruction cache fetch address and provides said first target address prediction in response to the fetch address.

24. (currently amended) The apparatus of claim 23, wherein said first branch predictor provides said first target address prediction in response to the fetch address whether or not a ~~unconditional~~ branch instruction is present in a third line of the instruction cache, said third instruction cache line selected subsequent to selection of said first instruction cache line.

25. (currently amended) The apparatus of claim 23, wherein said first branch predictor provides said first target address prediction prior to said instruction decode logic decoding said ~~unconditional~~ branch instruction.

26. (original) The apparatus of claim 1, wherein said first branch predictor comprises a branch target address cache indexed by the instruction cache fetch address.

27. (original) The apparatus of claim 1, wherein said first branch predictor comprises a speculative call/return stack.

28. (currently amended) A branch prediction apparatus in a processor, comprising:

first and second branch predictors, for making first and second predictions of a target address of ~~an unconditional~~ branch instruction;

comparison logic, coupled to said first and second branch predictors, configured to provide a comparison of said first and second predictions;

instruction decode logic, configured to decode said ~~unconditional~~ branch instruction and to generate a type thereof; and

control logic, coupled to said instruction decode logic, for causing the processor to branch based on said first prediction;

wherein said control logic selectively overrides, based on said type of said branch instruction and said comparison, said first prediction with said second prediction.

29. (original) The apparatus of claim 28, wherein said instruction decode logic decodes and generates said branch instruction type subsequent to said first branch predictor making said first prediction.

30. (original) The apparatus of claim 28, wherein said second branch predictor makes said second prediction in response to said instruction decode logic decoding said branch instruction.

31. (currently amended) The apparatus of claim 28, ~~wherein said second branch predictor comprises~~ further comprising:

wherein said first branch predictor is further configured to provide a first direction prediction of said branch instruction, prior to said instruction decode logic decoding said branch instruction and generating said type thereof, said first direction prediction for predicting whether said branch instruction will be taken or not taken; and

a branch history table for providing a second direction prediction based on said type of said branch instruction ~~in said second prediction~~, said second direction prediction for predicting whether said branch instruction will be taken or not taken.

32. (currently amended) The apparatus of claim 28, wherein if said branch type is a conditional branch type, said comparison logic compares first and second direction predictions ~~in said first and second predictions~~, respectively.

33. (original) The apparatus of claim 32, wherein if said first and second direction predictions do not match, said control logic overrides said first prediction with said second prediction.

34. (original) The apparatus of claim 28, wherein said comparison logic compares first and second target addresses in said first and second predictions, respectively.

35. (original) The apparatus of claim 34, wherein said control logic selectively overrides, based on said type of said branch instruction, said first prediction with said second prediction by causing the processor to branch to said second target address if said first and second target addresses do not match.

36. (currently amended) A pipelined processor, comprising:

a speculative branch predictor, for making a speculative prediction of ~~an~~ ~~unconditional~~ branch instruction target address;

control logic, coupled to said speculative branch predictor, for branching the processor based on said speculative target address prediction;

instruction decode logic, configured to decode and generate a type of said branch instruction; and

a non-speculative branch predictor, coupled to said instruction decode logic, for making a non-speculative prediction of said ~~unconditional~~ branch instruction target address;

wherein said control logic subsequently selectively branches the processor based on said non-speculative target address prediction and said branch instruction type.

37. (original) The processor of claim 36, further comprising:

an instruction cache, coupled to an address bus for receiving a fetch address, said fetch address selecting a line of instructions for provision to said instruction decode logic.

38. (original) The processor of claim 37, wherein said speculative branch predictor makes said speculative prediction even though a possibility exists that no branch instruction is present in said line of instructions.

39. (currently amended) The processor of claim 36, wherein said non-speculative branch predictor makes said non-speculative target address prediction in response to said instruction decode logic decoding said ~~unconditional~~ branch instruction.

40. (original) A pipelined processor, comprising:

a branch target address cache, for providing a speculative target address of an instruction prior to decoding of said instruction;

a target address calculator, for calculating a non-speculative target address of said instruction after said decoding of said instruction; and

a comparator, coupled to said branch target address cache and said target address calculator, for comparing said speculative and non-speculative target addresses;

wherein said processor branches to said speculative target address, wherein the processor subsequently branches to said non-speculative target address if said speculative and non-speculative target addresses miscompare and if said instruction is a type comprised in a first set of instruction types.

41. (original) The processor of claim 40, wherein said first set of instruction types includes a return instruction type.

42. (original) The processor of claim 40, wherein said first set of instruction types includes a program counter-relative branch instruction type.

43. (original) The processor of claim 40, wherein said first set of instruction types includes a conditional branch instruction type.

44. (canceled)

45. (currently amended) A method for branching in a pipelined processor, comprising:

generating a speculative target address of ~~an unconditional~~ branch instruction;

branching the processor to said speculative target address;

decoding said ~~unconditional~~ branch instruction after said branching;

generating a non-speculative target address of said ~~unconditional~~ branch instruction after said decoding;

determining a branch type of said ~~unconditional~~ branch instruction;

determining whether said speculative and non-speculative target addresses match; and

selectively branching, based on said branch type, to said non-speculative target address if said speculative and non-speculative target addresses do not match.

46. (original) The method of claim 45, further comprising:

determining if said branch type is a program counter-relative type branch instruction;

wherein said selectively branching comprises branching to said non-speculative target address if said speculative and non-speculative target addresses do not match and if said branch type is a program counter-relative type branch instruction.

47. (original) The method of claim 45, further comprising:

determining if said branch type is a return type branch instruction;

wherein said selectively branching comprises branching to said non-speculative target address if said speculative and non-speculative target addresses do not match and if said branch type is a return type branch instruction.

48. (currently amended) The method of claim 45, further comprising:

determining if said branch type is ~~an indirect~~ conditional type branch instruction;

wherein said selectively branching comprises branching to said non-speculative target address if said speculative and non-speculative target addresses do not match and if said branch type is ~~an indirect~~ conditional type branch instruction.

49. (original) The method of claim 45, further comprising:

determining if said branch type is an indirect type branch instruction;

wherein said selectively branching comprises not branching to said non-speculative target address if said branch type is an indirect type branch instruction.

50. (original) The method of claim 45, further comprising:

generating a non-speculative direction prediction of said branch instruction after said decoding said branch instruction; and

branching to a next sequential instruction pointer after said branch instruction if said non-speculative direction prediction indicates said branch instruction will not be taken.

51. (new) A method for processing a branch instruction in a pipelined microprocessor having an instruction cache, the method comprising:

fetching a cache line containing the branch instruction from the instruction cache;

generating a first prediction of a target address of the branch instruction concurrently with said fetching;

branching instruction fetching to the first target address prediction;

determining a type of the branch instruction, after said branching to the first target address prediction, wherein the type is one of a plurality of predetermined branch instruction types;

generating a second prediction of the target address of the branch instruction based on the type of the branch instruction, after said determining the type of the branch instruction; and

if the first target address prediction does not match the second target address prediction:

if the type of the branch instruction is a first of the plurality of predetermined types:

overriding said branching to the first target address prediction by branching instruction fetching to the second target address prediction; and

if the type of the branch instruction is a second of the plurality of predetermined types:

foregoing overriding said branching to the first target address prediction.

52. (new) The method as recited in claim 51, wherein the second of the plurality of predetermined types is an indirect branch instruction type.

53. (new) The method as recited in claim 51, wherein the first of the plurality of predetermined types is a program counter-relative branch instruction type.

54. (new) The method as recited in claim 51, wherein the first of the plurality of predetermined types is a return branch instruction type.

55. (new) The method as recited in claim 51, wherein the first of the plurality of predetermined types is a conditional branch instruction type.

56. (new) The method as recited in claim 55, further comprising:

generating a first prediction of a direction of the branch instruction, wherein said branching instruction fetching to the first target address prediction is performed only if the first direction prediction predicted the branch instruction will be taken;

generating a second prediction of the direction of the branch instruction, after said determining the type of the branch instruction; and

if the type of the branch instruction is the conditional branch instruction type:

determining whether the first direction prediction matches the second direction prediction; and

if the first direction prediction does not match the second direction prediction:

if the second direction prediction predicted the branch instruction will be taken:

overriding said branching to the first target address prediction by branching instruction fetching to the second target address prediction; and

if the second direction prediction predicted the branch instruction will not be taken:

overriding said branching to the first target address prediction by branching instruction fetching to a next instruction sequential to the branch instruction.

57. (new) The method as recited in claim 51, wherein said generating a first prediction of a target address of the branch instruction is performed without knowing whether the branch instruction is actually present in the cache line.

58. (new) The method as recited in claim 51, further comprising:

storing a history of target addresses of previously executed branch instructions, prior to said fetching a cache line containing the branch instruction, wherein said generating the first prediction of the target address is performed only if the history contains a target address for the branch instruction; and

branching instruction fetching to the second target address prediction, if the history does not contain a target address for the branch instruction.

59. (new) The method as recited in claim 58, further comprising:

generating a prediction of a direction of the branch instruction, if the type of the branch instruction is a conditional branch instruction type; and

said branching instruction fetching to the second target address prediction, only if the direction prediction predicted the branch instruction will be taken.

60. (new) The method as recited in claim 51, further comprising:

executing the branch instruction to determine a correct target address of the branch instruction, after said branching to the second target address prediction, wherein the correct target address is not a prediction; and

branching instruction fetching to the correct target address, if the second target address does not match the correct target address.

61. (new) A pipelined microprocessor having an instruction cache, comprising:

instruction fetch logic, configured to fetch a first cache line containing a branch instruction from the instruction cache;

a first branch predictor, coupled to said instruction fetch logic, configured to generate a first prediction of a target address of the branch instruction concurrently with fetching the first cache line;

wherein said instruction fetch logic is configured to fetch a second cache line from the instruction cache at the first target address prediction after fetching the first cache line;

instruction decode logic, coupled to said instruction fetch logic, configured to determine a type of the branch instruction after the first branch predictor generates the first prediction, wherein the type is one of a plurality of predetermined branch instruction types;

a second branch predictor, coupled to said instruction decode logic, configured to generate a second prediction of the target address of the branch instruction if the type of the branch instruction is a first of the plurality of predetermined types;

a third branch predictor, coupled to said instruction decode logic, configured to generate the second prediction of the target address of the branch instruction if the type of the branch instruction is a second of the plurality of predetermined types; and

comparison logic, coupled to said instruction fetch logic, for comparing the first and second target address predictions;

wherein if the first target address prediction does not match the second target address prediction:

if the type of the branch instruction is the first of the plurality of predetermined types:

said instruction fetch logic is configured to override the first branch predictor by fetching a third cache line from the instruction cache at the second target address prediction; and

whereas if the type of the branch instruction is the second of the plurality of predetermined types:

said instruction fetch logic is configured to forego overriding the first branch predictor.

62. (new) The microprocessor as recited in claim 61, wherein the second of the plurality of predetermined types is an indirect branch instruction type, wherein said third branch predictor comprises a branch predictor for predicting target addresses of indirect branch instructions.

63. (new) The microprocessor as recited in claim 61, wherein the first of the plurality of predetermined types is a program counter-relative branch instruction type, wherein said second branch predictor comprises a branch predictor for predicting target addresses of program counter-relative branch instructions.

64. (new) The microprocessor as recited in claim 61, wherein the first of the plurality of predetermined types is a return branch instruction type, wherein said second branch predictor comprises a branch predictor for predicting target addresses of return instructions.

65. (new) The microprocessor as recited in claim 61, wherein the first of the plurality of predetermined types is a conditional branch instruction type, wherein said second branch predictor comprises a branch predictor for predicting target addresses of conditional branch instructions.

66. (new) The microprocessor as recited in claim 65, further comprising:

wherein said first branch predictor is further configured to generate a first prediction of a direction of the branch instruction, wherein said instruction fetch logic fetches the second cache line from the instruction cache at the first target address prediction only if the first direction prediction predicted the branch instruction will be taken;

a fourth branch predictor, configured to generate a second prediction of the direction of the branch instruction, if the type of the branch instruction is the conditional branch instruction type; and

second comparison logic, coupled to said instruction fetch logic, configured to compare the first and second direction predictions;

wherein if the type of the branch instruction is the conditional branch instruction type:

if the first direction prediction does not match the second direction prediction:

if the second direction prediction predicted the branch instruction will be taken:

said instruction fetch logic is configured to override the first branch predictor by fetching a third cache line

from the instruction cache at the second target address prediction; and

whereas if the second direction prediction predicted the branch instruction will not be taken:

said instruction fetch logic is configured to override the first branch predictor by fetching a third cache line from the instruction cache containing a next instruction sequential to the branch instruction.

67. (new) The microprocessor as recited in claim 61, wherein said first branch predictor generates the first target address prediction without knowing whether the branch instruction is actually present in the first cache line.

68. (new) The microprocessor as recited in claim 61, wherein said first branch predictor comprises:

a branch target address cache (BTAC), configured to store a history of target addresses of previously executed branch instructions, prior to fetching the first cache line containing the branch instruction, wherein said BTAC generates the first prediction of the target address only if the history contains a target address for the branch instruction; and

wherein said instruction fetch logic is configured to fetch the third cache line from the instruction cache at the second target address prediction, if the history does not contain a target address for the branch instruction.

69. (new) The microprocessor as recited in claim 68, wherein said BTAC is further configured to generate a prediction of a direction of the branch instruction if the type of the branch instruction is a conditional branch instruction type, wherein said instruction fetch logic is configured to fetch the second cache line from the instruction cache at the second target address prediction, only if the direction prediction predicted the branch instruction will be taken.

70. (new) The microprocessor as recited in claim 61, further comprising:

an execution unit, coupled to said instruction fetch logic, configured to execute the branch instruction to determine a correct target address of the branch instruction, after said instruction fetch logic branches to the second target address prediction, wherein the correct target address is not a prediction; and

wherein said instruction fetch logic is further configured to fetch a fourth cache line from the instruction cache at the correct target address, if the second target address does not match the correct target address.